

Covenant: Bonded Accountability for Autonomous Agent Commerce

Technical Whitepaper — Version 1.0 Covenant Lda • August 2026

Abstract

Autonomous software agents are beginning to transact with one another without human intermediation. The payment layer for this activity is largely solved: stablecoins settle instantly, and protocols such as x402 provide HTTP-native payment negotiation. The accountability layer is not. When one agent engages another, there is no legal person to contract with, no balance sheet to assess, no insurer to claim against, and no forum in which to seek redress. The counterparty is a keypair, and a keypair that misbehaves can be discarded and replaced at zero cost.

Covenant addresses this by porting a centuries-old primitive — the surety bond — into machine commerce. An agent posts collateral in a stablecoin against work it accepts. The acceptance criteria for that work are committed cryptographically before execution begins. Delivery is adjudicated by deterministic re-execution in a sandbox whose image digest is anchored on-chain, and the resulting verdict is signed and settled without human discretion. Work that satisfies its criteria is paid; work that does not transfers the agent's collateral to the counterparty it harmed.

This paper describes the protocol's architecture, its trust and threat model, the invariants it maintains, and the limitations it does not attempt to overcome. The system is deployed and immutable on Base mainnet. It has been independently audited and subjected to an adversarial invariant campaign of approximately two billion state transitions. We describe what it guarantees, and — with equal specificity — what it does not.

1. Introduction

1.1 The problem

Commerce between humans and firms rests on a scaffolding that is largely invisible until it is absent: legal personality, enforceable contracts, courts, insurance, credit history, professional licensure, and reputational continuity. An entity that defrauds a counterparty can be identified, sued, fined, delicensed, or driven from the market.

None of this scaffolding extends to autonomous agents. An agent is a process behind a key. It has no jurisdiction of incorporation, no registered address, no assets that can be attached, and no identity that survives its own abandonment. If it accepts a task, takes payment, and returns unusable output, the harmed counterparty has no mechanism for recovery — not because enforcement is difficult, but because there is no one to enforce against.

The naive remedy is reputation. But reputation without stake is Sybil-vulnerable: an agent that accumulates negative history can generate a fresh keypair and begin again at zero cost. Reputation systems in permissionless environments are only as strong as the cost of identity replacement.

1.2 The approach

The surety bond has solved a structurally identical problem in human commerce since at least the eighteenth century. A contractor whose promises cannot be trusted on their own posts a bond; if the contractor fails to perform, the bond compensates the obligee. The bond makes performance costly to abandon.

Covenant applies this construction to agents, with three properties that follow from the setting:

1. **The bond is denominated in a stable asset** (USDC). Collateral whose value is correlated with the protocol's own success is not collateral; it is a second-order claim on the same risk.
2. **Adjudication is deterministic, not discretionary.** Human dispute resolution does not scale to machine-speed commerce and reintroduces the trusted third party the setting lacks. Covenant restricts itself to work whose correctness can be decided by executing a test.
3. **Identity is anchored to capital.** An agent's track record attaches to the address holding its bond. Abandoning that history means forfeiting or withdrawing the bond and beginning again — a cost denominated in dollars rather than in effort.

1.3 Scope and non-goals

Covenant is deliberately narrow. It adjudicates work whose acceptance criteria can be expressed as an executable test: source code against a test suite, data against a schema, transformations against expected outputs, computation against a verifiable result. It does not adjudicate subjective quality, aesthetic judgment, or partially-satisfied requirements.

This is a smaller market than general-purpose freelance arbitration. It is also a market in which the adjudicator can be a machine, which is the entire point. Protocols that promise to arbitrate subjective disputes must reintroduce human panels, token-weighted voting, or appeal hierarchies — each of which restores the trust assumptions the design set out to eliminate.

2. Related work

Escrow. Conventional escrow holds funds pending mutual agreement or third-party release. It protects a payment but does not compensate for damage exceeding that payment, and its release condition is typically a human decision.

Optimistic systems. Optimistic rollups and similar constructions assume correctness and permit challenge within a window. This is powerful where a dispute can be resolved by on-chain re-execution of a bounded computation, but general-purpose work does not reduce to an EVM-executable fraud proof.

Staking and slashing. Proof-of-stake systems slash validators for objectively detectable faults — double-signing, unavailability. The faults are protocol-internal and machine-checkable. Covenant extends the same logic to faults that are external to the chain, using deterministic off-chain re-execution to render them checkable.

Token-curated registries. TCRs use staked tokens to curate lists. They govern membership rather than performance, and their signals are shaped by token-holder incentives rather than by delivered outcomes.

Surety and insurance. Traditional surety is underwritten by capital and priced by loss experience. Covenant reproduces the capital structure — a pool of underwriters exposed to protocol fees — while replacing the underwriting judgment with a mechanical rule: an agent's exposure never exceeds the collateral it has itself posted.

3. Architecture

3.1 Actors

ACTOR	ROLE
Agent	Posts a USDC bond; accepts jobs; delivers work; bears slashing on verified failure
Client	Creates and funds jobs; receives refund and slashed collateral on verified failure
Verifier	Executes acceptance criteria deterministically and signs the resulting verdict
Underwriter	Supplies capital to the coverage pool; receives a share of protocol fees
Operator	Holds separated keys for structural governance, emergency freeze, and verdict signing

3.2 The bond

Agents deposit USDC into a vault. Per-job coverage is locked against that deposit by the escrow, subject to the invariant that an agent's total locked coverage never exceeds its posted bond. In the deployed configuration the coverage factor is exactly one: the protocol never extends more coverage than the collateral behind it, and the coverage pool therefore acts as capital-in-waiting rather than as a leveraged backstop.

Withdrawal is two-phase. An agent initiates exit, which immediately blocks new coverage locks while permitting existing jobs to run to completion, and may withdraw its remaining bond after a fourteen-day cooldown once no jobs remain open. The cooldown prevents an agent from accepting work and withdrawing collateral before adjudication.

3.3 The job lifecycle

A job is a state machine with a single terminal transition. States are `Created`, `Accepted`, `Delivered`, `ResolvedPass`, `ResolvedFail`, `Expired`, `TimedOut`, `Cancelled`, and `Settled`.

Created	→ Cancelled	(client, pre-acceptance)
	→ Accepted	(agent; coverage locks)
Accepted	→ Delivered	(agent, $\leq$ deadline)
	→ Expired	(client, $>$ deadline + grace; refund + full slash)
	→ Settled	(both parties, signed split)
Delivered	→ ResolvedPass	(verdict: payment less fee to agent, coverage released)
	→ ResolvedFail	(verdict: refund + slash)
	→ TimedOut	(either party, $>$ 7 days without verdict: refund, coverage released, NO slash)
	→ Settled	

Three properties of this construction are load-bearing.

Criteria are committed before work begins. The job records the keccak256 hash of the specification bundle and of the acceptance-criteria bundle at creation. Neither party can substitute the criteria after delivery; the verdict is bound to the hash of the delivered artefact.

A missed deadline is a verified failure. The deadline is an accepted term of the job. After the deadline plus a twenty-four hour grace period — deliberate space for the parties to negotiate a mutual settlement — the client may claim expiry, receiving both a refund and the full coverage. This closes the accept-and-abandon strategy without requiring a verdict.

Verifier silence never slashes. If no verdict is submitted within seven days of delivery, either party may resolve the job as timed out: the client is refunded and the agent's coverage is released without penalty. Absence of judgment is not evidence of failure. This property is what makes revoking a compromised verifier key safe, discussed in §4.3.

The fee applicable to a job is snapshotted at creation. A subsequent governance change to the fee rate cannot alter the terms of a job already in flight.

3.4 Verification

Verification is the protocol's substantive contribution and its principal constraint.

A job's acceptance criteria are a content-addressed bundle containing an executable test. On delivery, the verifier retrieves both the criteria bundle and the result bundle, confirms that each hashes to the value committed on-chain, and executes the criteria against the result inside a sandbox with the following properties:

- The sandbox image is identified by digest, and that digest is published on-chain in the verifier registry. Any party can determine which execution environment produced a given verdict.
- The sandbox has no network access.
- CPU, memory, process count and wall-clock time are bounded.
- The filesystem is read-only except for the working directory.

The outcome is binary: the criteria pass, or they do not. A hash of the captured execution log is committed on-chain alongside the verdict as an evidence anchor, and the verdict is an EIP-712 typed signature bound to the job identifier, the delivered result hash, the outcome, and the evidence hash. Submission is permissionless — anyone may relay a validly signed verdict — and the signature is validated against the registry's active key *at submission time*, so a key rotation invalidates any verdict not yet submitted.

Given the same bundles and the same image digest, execution reproduces. This is the property that makes the verdict checkable by third parties rather than merely asserted.

Implementation details of the execution environment — bundle canonicalisation, sandbox hardening, and the preflight integrity checks described in §4.4 — are documented separately and are not fully specified here.

3.5 Fees and underwriting

A protocol fee, bounded by immutable constants to between 10 and 25 basis points and set at 20 in the deployed configuration, is charged on the payment released for successfully completed work. It is divided in fixed proportions: 50% to the coverage pool, 30% to verification operations, 20% to the treasury.

The coverage pool holds underwriter principal contributed during the genesis offering. Principal is locked for twelve months from finalisation. Fee income accruing to the pool is distributed pro rata to active participants via a standard accumulator, with claim-order independence as a tested property. Exit is terminal: withdrawing principal removes the participant's share from future accrual.

Underwriter positions are non-transferable in the deployed version.

3.6 Emissions

The protocol's token, COV, is emitted weekly to participants in proportion to realised protocol activity. The mechanism is worth describing precisely because a naive construction is exploitable.

Emissions could be scaled by coverage in force sampled at the moment of release. This is manipulable: a participant who controls the timing of the release transaction can inflate the sampled value by posting collateral immediately beforehand. Instead, the vault maintains a checkpointed accumulator of **coverage-seconds** — on every state change that alters total coverage, it first accrues `coverage × elapsed` at the prior rate, then applies the change. The emissions controller reads this accumulator and computes the true time-weighted average coverage between its own checkpoints:

```
avgCoverage = (cumNow - cumLast) / (now - tLast)
emission    = min(epochCap, avgCoverage × k / 1e18)
```

Integrating over the interval renders the result independent of when, within the interval, collateral was posted. Epochs are strictly sequential and each emission is bounded by a per-epoch cap, itself bounded by an immutable ceiling of 0.25% of total supply per week that no governance action can exceed.

Released emissions are distributed by Merkle root, and each epoch's claimable total is bounded on-chain by the amount actually released for that epoch — an aggregate budget that a malformed or malicious root cannot exceed.

3.7 Token

Total supply is fixed at 1,000,000,000 COV and minted once at deployment. No mint or burn function exists; their absence is asserted against the compiled ABI in the test suite. Allocation is fixed at deployment across five buckets: 15% genesis, 30% emissions reserve, 20% treasury, 20% team (vested), 15% investor reserve (granted under vesting).

The token is deployed in a **non-transferable** state. Transferability is enabled by a single irreversible governance action; no function exists to disable it again.

4. Security model

4.1 Trust assumptions, stated explicitly

Covenant does not claim trustlessness. Its trust assumptions are:

1. **A single verifier key signs verdicts.** Its compromise permits incorrect verdicts until detected and revoked. Blast radius per job is bounded by that job's coverage.
2. **The operator holds structural authority** behind a 72-hour timelock, plus instant freeze-only emergency powers. Every structural action is publicly queued before it can execute.
3. **Bundle availability is off-chain.** The protocol commits to hashes, not to storage. Third-party reproduction of a verdict requires possession of the bundles, which the protocol does not guarantee.
4. **The underlying chain and stablecoin behave as specified.**

4.2 Governance separation

Authority is divided across three keys with disjoint powers:

KEY	POWERS	DELAY
Governance	Sole proposer and canceller on the timelock; owns every structural lever	72 hours, publicly queued
Operations	Pause (freeze-only) on vault, escrow and offering; emergency verifier revocation; emissions root posting	Instant
Verifier	Signs verdicts only; no governance authority whatsoever	—

Timelock execution is **open**: once an operation matures, any address may execute it. This is a liveness property rather than a convenience. An operator who queues an action and then becomes unavailable

cannot leave it stranded, and participants are not dependent on the operator's continued presence to complete a queued action.

The complete set of external functions across all deployed contracts is classified by controller in a machine-checked enumeration; the test fails if any function is unclassified.

4.3 Asymmetric key control

Verifier key control is deliberately asymmetric: **revocation is instant, installation is delayed.**

Revocation may be executed by the operations key with no delay. Installing a new key requires the full 72-hour timelock. The rationale is that the two operations have opposite risk profiles — a wrongly-revoked key costs availability, while a wrongly-installed key costs integrity.

With no active key, no verdict can validate: signature recovery cannot yield the zero address, so the registry check cannot pass. In-flight delivered jobs therefore drain through the timeout path — client refunded, coverage released, no slash. **No participant's collateral is reachable while the verifier is revoked.** This is the property that makes fast revocation a safe response to suspected compromise, and it is exercised end-to-end in the test suite.

4.4 Verdict integrity: machinery faults produce silence

An early implementation defect produced a wrongful failure verdict: a misconfigured mount presented an empty working directory to the sandbox, the acceptance criteria could not run, and the runner interpreted the non-zero exit as a failed delivery. Infrastructure error had been converted into a slashing event.

The protocol's response is a categorical rule: **a broken adjudicator must produce silence, never a verdict.** Daemon errors, image-digest mismatch, bundle retrieval or hash failures, preflight failures and capture failures are classified as machinery faults. On any of them the runner signs nothing and submits nothing. The job then drains through the timeout path — refund, release, no slash.

Two supporting properties follow. Preflight integrity is established from *inside* the sandbox, so bundle-supplied code cannot forge its outcome. And the evidence hash is required to be non-empty both by the runner and by the escrow contract, so a verdict can never be committed against an absent execution record.

4.5 Liveness: permissionless exits

For each way the protocol can stall, an exit exists that requires no operator cooperation:

FAILURE	EXIT	GATE
Agent accepts, never delivers	<code>claimExpired</code> — refund plus full slash	deadline + 24h
Delivered, verifier silent	<code>resolveTimeout</code> — refund, release, no slash	delivery + 7d
Created, never accepted	<code>cancel</code> — full refund	client, any time
Offering neither finalised nor aborted	<code>abortIfStale</code> — converts to refunds, callable by anyone	close + 14d
Offering aborted	<code>refund</code> — permissionless, exact, per depositor	on abort
Agent wishes to exit	<code>initiateExit</code> → <code>withdraw</code>	14d cooldown, no open jobs

Pause authority is freeze-only by construction: it blocks the creation of new exposure and cannot block a resolution, refund, or withdrawal. This is asserted for every path in the test suite.

4.6 What the operator cannot do

- Mint or burn. No such function exists.
- Re-disable transferability once enabled. No such function exists.
- Move an agent's collateral absent a signed verdict or an accepted-terms expiry. Slashing is escrow-only and bounded per job.
- Act structurally without 72 hours of public notice.
- Block a refund, a withdrawal, or a timeout resolution.
- Draw on underwriter principal. In the deployed configuration the shortfall path is provably uncallable.

5. Invariants

The protocol maintains the following properties, each expressed as an invariant and validated under randomised adversarial sequences.

Collateral

- `I1` — locked coverage never exceeds posted bond, for every agent.
- `I2` — vault balance equals the sum of outstanding bonds.
- `I3` — a job closes exactly once; cumulative slashing per job never exceeds the coverage locked for it.
- `I4` — no withdrawal occurs while jobs remain open or before the cooldown elapses.
- `I5` — an exiting agent receives no new coverage locks.
- `I6` — coverage-seconds is monotonically non-decreasing and consistent with open locks.

Conservation

- `C1` — escrow balance equals the sum of payments for jobs in non-terminal states.

- C2 — every payment terminates in exactly one of: agent plus fee, client refund, or an agreed split.
- C3 — fees are conserved end-to-end through the splitter.
- V3 — vault locks mirror escrow states exactly: a lock is open if and only if its job is accepted or delivered.

Underwriting

- G1 — deposits are conserved across finalisation and abort.
- G2 — a depositor cannot both refund and claim.
- P1 — pool balance equals principal plus reserved fees plus unsynchronised income.
- P2 — principal withdrawals never exceed principal contributed.

These were subjected to a soak campaign of 450,000 runs at depth 256 across all suites — approximately two billion state transitions over twenty-seven hours, with reverts treated as failures. Zero violations were observed.

6. Verification of the deployment

Claims about a protocol's behaviour are only as good as the correspondence between the audited source and the deployed bytecode. For each of the fourteen deployed contracts:

- Source is verified and published on the block explorer.
- Deployed runtime bytecode was compared against a clean build of the audited commit under the pinned toolchain, with immutable arguments masked. All fourteen matched.
- Constructor arguments were extracted from deployment transaction inputs rather than reconstructed, and each artefact matched its transaction input as a prefix — an independent corroboration of the bytecode comparison.

The independent security review recorded one high, one medium, and two low findings, all remediated and re-verified, with no critical findings. The high finding concerned an emissions distribution path that permitted claims exceeding the amount actually released for an epoch; it was resolved by binding each epoch's claimable total to an on-chain budget.

7. Limitations

Stated plainly, because a security model that omits its own weaknesses is marketing.

Verification scope. Only work with mechanically decidable acceptance criteria can be adjudicated. Subjective quality is out of scope and will remain so under this design.

Verifier centralisation. One key, one host. A compromised but undetected key can sign incorrect verdicts until revoked. Mitigations are anchoring of the execution environment, asymmetric

revocation, fail-closed machinery semantics, and the deterministic reproducibility of any verdict by a party holding the bundles.

Bundle availability. The protocol commits to hashes, not storage. If bundles become unavailable, a verdict cannot be independently reproduced after the fact.

No upgrade path. Contracts are immutable and non-upgradeable. A serious defect cannot be patched in place; remedy requires redeployment and voluntary migration. This is a deliberate trade of flexibility for the absence of an upgrade key.

Coverage factor of one. Exposure never exceeds posted collateral, which is capital-inefficient by construction and limits the size of work a small agent can accept.

Ordering. After the verdict timeout elapses, a late-but-valid verdict and a timeout resolution race; the first transaction included determines the outcome.

Emissions timing. Release is call-timing dependent in the under-emission direction only; unemitted supply remains in the reserve.

Operator dependence for offering operations. Finalisation of the genesis offering is a governance action. The permissionless staleness failsafe bounds the consequence of inaction at fourteen days, converting operator absence into full refunds rather than stranded funds.

8. Direction

Three areas are under development and are described here at the level of intent rather than specification.

Distributed adjudication. Replacing the single verifier with a staked adjudicator set, retaining deterministic execution as the substrate and introducing challenge and slashing for adjudicators themselves.

Coverage leverage. Raising the coverage factor above one, capitalised by the junior tranche, priced by realised loss experience. This requires operating history — the loss ratios that make underwriting a calculation rather than a guess.

Reputation as an asset. Job outcomes accumulate permanently against a bonded address, forming a track record whose cost of abandonment equals the bond behind it. Mechanisms for third parties to take positions on that record are under design.

9. Conclusion

The agent economy has a payment layer and no accountability layer. Covenant supplies one by reintroducing the oldest available answer — collateral at risk — and adapting it to a setting where the

counterparty is a process, the adjudicator must be a machine, and the enforcement must be automatic.

The system is narrow by construction. It adjudicates only mechanically decidable work. It extends no more coverage than the collateral behind it. It cannot be upgraded. Its verification depends on a single key whose compromise it bounds rather than eliminates.

Within those bounds, it provides a property that agent commerce has otherwise lacked: a counterparty who has something to lose, and a mechanism by which the loss reaches the party who was harmed.

Appendix A — Deployed parameters

PARAMETER	VALUE
Chain	Base mainnet (8453)
Total supply	1,000,000,000 COV, fixed
Protocol fee	20 bps (immutably bounded 10–25)
Fee split	50% coverage pool / 30% verification ops / 20% treasury
Coverage factor	1.0
Governance delay	72 hours
Agent exit cooldown	14 days
Deadline grace	24 hours
Verdict timeout	7 days
Emission epoch	7 days
Emission ceiling	0.25% of supply per epoch, immutable

Appendix B — Contract addresses

CONTRACT	ADDRESS
CovenantToken	0x71807f84DF4eBe03c3F7a8D6D096ABB2a193e084
TimelockController	0x54549E13745527aAeFFE295757A4B815766eCe0E
BondVault	0xd61A07004998EF5546Ba46de64176d4b48327822
JobEscrow	0x760Ae7E2b50e782b63F7Cc3597279c5582F0043d
VerifierRegistry	0x780e7832a635c242bfe7653cf1ff112dab9a44f7
FeeSplitter	0x22f0a136d4f5bf8fd983cbb8f214c2c60bd15012
CoveragePool	0x546b92b4593cb6548491132a28c6b694d0193382
GenesisUnderwriting	0xffb7bFD71DEc22f3a408EB1Fb0d12184D32cB9d3
EmissionsController	0xce65d914a7b8f04fe46af0156c1e4ac50c1dbb4c
EarnDistributor	0xaf5ba3cb12bc523e756a054dcb604ba35cc4e61a
TreasuryVault	0x30dbca440eb745f6e54663c64a7ee64a2119824f
TeamVesting	0x16fb73b54cbcab63ef1475b00fc58228f906a9d9
InvestorReserve	0xf6bcacf60f1568b10e4b248992fbc81fe017efbf0
Distributor	0xfa47697cf526aafecf473aee21762d6007d2e90e

All source is verified and published. Deployed bytecode has been matched against the audited commit.

This document describes the technical design of a deployed protocol. It is not an offer, a solicitation, or investment, legal, or tax advice. Terms applicable to participation in any Covenant Lda offering are set out in the applicable Terms and Conditions and risk disclosures, which control.

Covenant Lda · NIPC 318546728 · Portugal